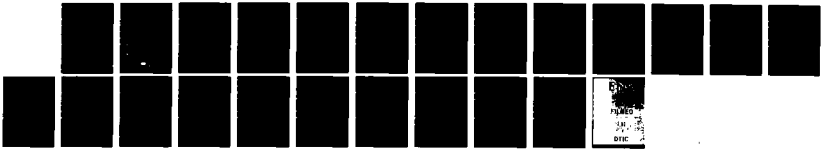
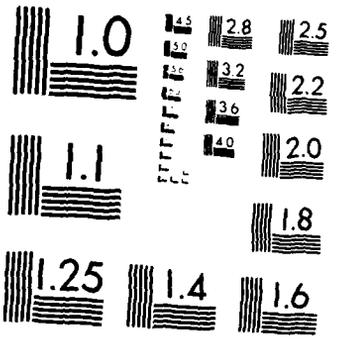
AD-A129 865 TRAFFIC ASSIGNMENT IN COMMUNICATION SATELLITES(U) 1/1
CARNEGIE-MELLON UNIV PITTSBURGH PA MANAGEMENT SCIENCES
RESEARCH GROUP E BALAS ET AL. APR 83 MSRR-491
UNCLASSIFIED N00014-82-K-0329 F/G 17/2 NL





TRAFFIC ASSIGNMENT IN COMMUNICATION SATELLITES

by

Egon Balas

and

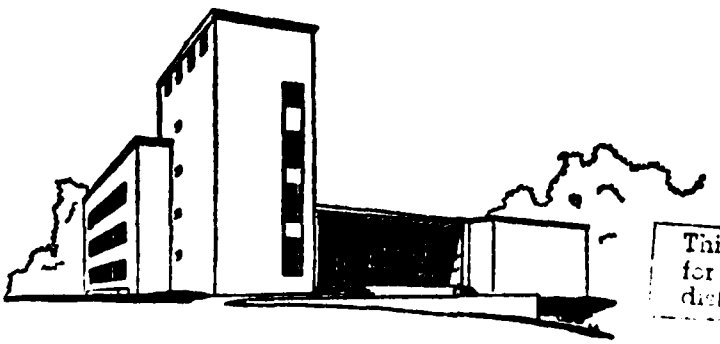
Philip R. Landweer

Carnegie-Mellon University

PITTSBURGH, PENNSYLVANIA 15213

GRADUATE SCHOOL OF INDUSTRIAL ADMINISTRATION

WILLIAM LARIMER MELLON, FOUNDER



W.P.#58-82-83

Management Science Research Report No. MSRR 491

TRAFFIC ASSIGNMENT IN COMMUNICATION SATELLITES

by

Egon Balas

and

Philip R. Landweer

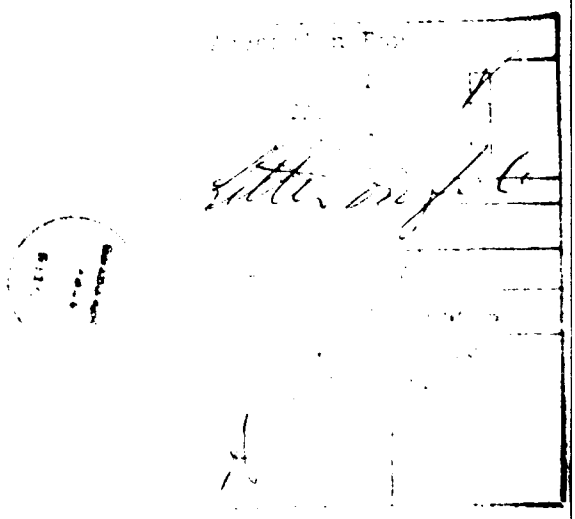
April 1983

The research underlying this report was supported by Grant ECS-8205425 of the National Science Foundation and Contract N00014-82-K-0329 NR 047-607 with the U.S. Office of Naval Research. Reproduction in whole or in part is permitted for any purpose of the U.S. Government.

Management Science Research Group
Graduate School of Industrial Administration
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

A high capacity communication satellite interconnects scores of ground stations simultaneously. Under the Satellite-Switched/Time Division Multiple Access (SS/TDMA) system, each channel of the satellite is allocated to a pair of ground stations for a certain time period, after which the whole set of allocations (called a switch) is changed simultaneously. The problem we address is to minimize the time length of the entire sequence of switches, subject to a limit on the number of switches. We formulate this as a 3-index bottleneck-sum assignment problem, and solve it by a heuristic that obtains consistently better results than earlier methods based on different formulations.



TRAFFIC ASSIGNMENT IN COMMUNICATION SATELLITES

by

Egon Balas

and

Philip R. Landweer

1. Introduction: Background and History

Every day a rapidly increasing volume of long distance TV, radio and telephone communications is transmitted digitally via satellites. A high capacity communication satellite interconnects simultaneously scores of transmitting and receiving stations. One of the advanced techniques for operating such a satellite communication network is the SS/TDMA (satellite-switched time-division multiple access) system, based on the use of highly directive spotbeam antennas [12], [1]. Under this scheme, each transponder on board of the satellite is allocated to a pair of zones (groups of ground stations) for a certain amount of time. Such a set of allocations involving all the transponders is called a switch mode. The allocations that make up a switch mode are changed simultaneously by an on-board switching facility. The whole sequence of switch modes that make up the schedule of allocations for a given time period is called a frame.

A major problem that arises in connection with SS/TDMA systems is the efficient scheduling of time slot allocations for a frame. To be more specific, the demand for a frame can be expressed as an nXn matrix $D = (d_{ij})$, the traffic matrix, where d_{ij} represents the length of the data burst from zone i to zone j, i.e., the amount of time for which a transponder needs

to be assigned to the pair of zones (i,j). Clearly, $d_{ij} \geq 0$ for all i,j. The scheduling task consists of allocating time slots on the satellite's transponders to pairs of zones. More specifically, it consists of decomposing the traffic matrix D into nxn switch matrices (or mode matrices) $D^k = (d_{ij}^k)$, k = 1,...,q, subject to certain feasibility conditions and satisfying some efficiency criterion. The feasibility conditions are as follows.

(1) For k = 1,...,q, every row and every column of D^k contains at most one nonzero element. In other words, in every switch mode, each transponder is assigned to at most one pair of zones.

$$(2) \qquad \sum_{k=1}^{q} D^k = D,$$

i.e., the q switch modes together meet total demand.

It is easy to see that for $q < n$ the problem is infeasible whenever D has at least one row or column whose entries are all positive. On the other hand, for q = n the problem amounts to 1-factorizing the complete bipartite graph $K_{n,n}$ (i.e., finding in $K_{n,n}$ n pairwise edge-disjoint perfect matchings), a problem whose feasibility is an immediate consequence of the Kőnig-Hall theorem on perfect matchings [7]. Thus for $q \geq n$ the problem is always feasible.

As to the efficiency criterion, a first objective is to satisfy total demand in a minimum amount of time. Since the time needed to transmit one switch mode is given by the largest entry of the switch matrix D^k, total transmission time for a given solution $(D^1,...,D^q)$ is

$$(3) \qquad T = \sum_{k=1}^{q} \max_{i,j} d_{ij}^{k}$$

Thus one approach is to address

Problem 1. [9] Find a schedule $(D^1,\ldots,D^q)$ that satisfies (1), (2) and minimizes (3).

The value of an optimal solution to this problem can be found out by inspection. First, it is not hard to see that any row sum or column sum of D is a lower bound on the value of T; hence the same is true of the maximum of these row and column sums, i.e., $T \geq T^*$, where

$$(4) \qquad T^* = \max\{\max_{i} \sum_{j=1}^{n} d_{ij}; \max_{j} \sum_{i=1}^{n} d_{ij}\}.$$

Second, it can be shown [9, 8] that (i) min $T = T^*$, (ii) there exists an optimal solution $(D^1,\ldots,D^q)$ with $q \leq n^2 - 2n + 2$; and (iii) this is the smallest bound on the value of q for which the existence of a schedule with $T = T^*$ can be guaranteed. Polynomial time procedures for finding an optimal solution to Problem 1 are given in [9, 8]. Although these procedures do not guarantee the minimality of q, theoretical considerations suggest that for a schedule $(D^1,\ldots,D^q)$ with $T = T^*$, the value of q is typically either equal to, or close to, $n^2 - 2n + 2 - z$, where z is the number of zeroes in D.

This takes us to the question of what is the significance of q, the number of switch modes within a frame. A schedule involving q switches requires as many switch reconfigurations. Further, if $q > n$, some data bursts must be split, i.e., only partially transmitted during a switch mode, with the rest transmitted in another switch mode. Switch reconfigurations and burst splittings entail the addition of preambles and

guard margins, not to mention other inconveniences. Therefore a second important objective is the minimization of q. As mentioned earlier, the smallest value of q for which feasible schedules exist, is n; but a schedule that is to achieve a total time of T*, usually requires a value of q of the order of n(n-2). Hence the second relevant problem in this context can be formulated as

Problem 2. [2] Find a schedule $(D^1,\ldots,D^q)$ that satisfies (1), (2) and q = n, and minimizes T.

The practical importance of Problem 2 comes from the fact that it avoids burst splittings. However, its relevance depends on how close its optimum is to T*. In this context, the (transmission) efficiency of a schedule with total transmission time T is defined as the ratio T*/T. Currently no bound is known on the efficiency of an optimal solution to Problem 2. Furthermore, as discussed below, Problem 2 is NP-complete and for realistic matrix sizes too large to be solved to optimality. Thus one has to look for approximate solution methods. Naturally, if no bound is known on the efficiency of an optimal solution, even less is known theoretically about the efficiency of approximate solutions. However, some empirical evidence exists in this respect. Camerini, Maffioli and Tartara [2] (in the following CMT) have developed and tested a heuristic for Problem 2, that consists of n successive applications of the following iterative step (which starts with k = 1):

Iterative Step. Solve the nxn assignment problem

$$\max \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_{ij}$$

$(AP)_k$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1,\ldots,n$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1,\ldots,n$$

$$x_{ij} \in \{0,1\}, \quad i,j = 1,\ldots,n$$

and, if x^* is the optimal solution found, let

$$d_{ij}^k = \begin{cases} d_{ij} & \text{if } x_{ij}^* = 1 \\ 0 & \text{if } x_{ij}^* = 0 \end{cases} \qquad i,j = 1,\ldots,n,$$

and set $d_{ij} \leftarrow -\infty$ if $x_{ij}^* = 1$, $d_{ij} \leftarrow d_{ij}$ if $x_{ij}^* = 0$, for $i,j = 1,\ldots,n$. Then if $k < n$, set $k \leftarrow k + 1$ and repeat the step. Otherwise $(D^1,\ldots,D^n)$ is the desired schedule.

The assignment problem is solved at each iteration by the Hungarian method [10] (for a more recent treatment, see [3] or [11]), whose time complexity is $O(n^3)$. Thus the time complexity of the whole procedure is $O(n^4)$. As to its transmission efficiency, on a set of 100 test problems with n = 20 and the entries of D integers drawn randomly from a uniform distribution over the interval [0, 100], it was found to be on the average about 89-90%, with a minimum of 83% and a maximum of 96%.

In a way, Problems 1 and 2 above represent two extremes, in which one of the two objectives of interest is driven to its minimum value without regard to what this entails in terms of the other. In order to examine some intermediate situations, CMT [2] have also addressed the following class.

Problem 3. For h = 1,2,..., find a schedule $(D^1,\ldots,D^q)$ that satisfies (1), (2) and q = hn, and minimizes T.

Finally, there is a substantial difference between burst splittings that do and do not separate bursts into nonadjacent time slots, the former (called proper burst splittings) being more costly than the latter. In order to reflect this differentiation, CMT [2] have formulated

Problem 4. For h = 1,2,..., find a schedule that satisfies (1), (2) and q = hn without proper burst splittings, and minimizes T.

CMT have modified their heuristic to obtain procedures of the same complexity (O(n4)) for finding approximate solutions to Problems 3 and 4. Ran on the same test problems, these procedures tended to produce solutions of improved efficiency as h was increased.

2. A New Heuristic for Problem 2

The CMT heuristic is based on the idea that for any k, maximizing the sum of elements of D^k will force into D^k as many of the larger elements of D as possible, thus reducing the value of the largest element left in D^{k+1}. In order to gain some insight into what might be a better heuristic, it is useful to look at some alternative formulations of Problem 3. The first formulation is a 3-index bottleneck-sum assignment problem with $3n^2$ constraints and n^3 variables, in which d_{ij}^k is represented as $d_{ij}x_{ijk}$, with $x_{ijk} = 0$ or 1:

$$\min_{x} \sum_{k=1}^{n} \max_{i,j} d_{ij} x_{ijk}$$

$$\text{(P2.1)} \qquad \begin{aligned} \sum_{j=1}^{n} x_{ijk} &= 1 && i,k = 1,\ldots,n \\ \sum_{i=1}^{n} x_{ijk} &= 1 && j,k = 1,\ldots,n \\ \sum_{k=1}^{n} x_{ijk} &= 1 && i,j = 1,\ldots,n \\ x_{ijk} &\in \{0,1\} && i,j,k = 1,\ldots,n \end{aligned}$$

The second formulation is a set partitioning problem with n^2 equations and n! variables, one variable associated with every nXn permutation matrix:

$$\text{(P2.2)} \qquad \begin{aligned} &\min cy \\ &Ay = e \\ &y_j \in \{0,1\},\ j = 1,\ldots,n! \end{aligned}$$

Here $e = (1,\ldots,1)^T$ has n^2 components, and every column a_j of A represents an nXn permutation matrix in vector form, i.e., $a_{ij} \in \{0,1\}$, $i = 1,\ldots,n^2$, with

$$\sum_{r=1}^{n} a_{(r-1)n+s,j} = 1, \qquad s = 1,\ldots,n$$

$$\sum_{s=1}^{n} a_{(r-1)n+s,j} = 1 \qquad r = 1,\ldots,n;$$

and all the columns of A are distinct. Further,

$$c_j = \max_{1 \le r,s \le n} a_{(r-1)n+s,j}\, d_{rs}.$$

Both (P2.1) and (P2.2) are NP-complete problems which one cannot hope to be able to solve to optimality except for small values of n. But they both offer some insights into the kind of heuristics that might be expected to work. In particular (P2.1) suggests that instead of solving a sequence of maximizing assignment problems $(AP)_k$, one might do better by solving a sequence of bottleneck, or min-max assignment problems. This leads the following:

Min-max Procedure.

Set k = 1 and go to the

Iterative Step. Solve the nXn bottleneck assignment problem

$$\min_{x} \max_{i,j} d_{ij} x_{ij}$$

$$(BAP)_k \qquad \sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1,\ldots,n$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1,\ldots,n$$

$$x_{ij} \in \{0,1\}, \qquad i,j = 1,\ldots,n,$$

and if x* is the optimal solution found, define

$$d_{ij}^{k} = \begin{cases} 1 & \text{if } x_{ij}^{*} = 1 \\ 0 & \text{if } x_{ij}^{*} = 0 \end{cases}, \qquad i,j = 1,\ldots,n$$

and set $d_{ij} \leftarrow \infty$ if $x_{ij}^{*} = 1$, $d_{ij} \leftarrow d_{ij}$ if $x_{ij}^{*} = 0$, for all i,j = 1,...,n.

Then if $k < n$, set $k \leftarrow k + 1$ and repeat the step. Otherwise $(D^1,\ldots,D^n)$ is the desired schedule.

On the other hand, the set partitioning formulation (P2.2) suggests the use of the greedy algorithm for set covering problems [4], modified to handle equations instead of inequalities. Since the column sum $|a_j|$ is the same for every column of A, and a cover can only contain pairwise orthogonal columns, the greedy choice rule is equivalent in this case to minimizing c_j. Thus the greedy algorithm adapted to (P2.2) amounts to sequentially choosing columns a_j of A in order of increasing c_j, while deleting after each choice all columns a_k such that $a_k a_j \geq 1$, until a set of n columns has been chosen. But in view of the definition of the a_j and c_j, this is precisely the Min-max Procedure stated above. Thus both problem formulations suggest the same heuristic approach.

The Iterative Step of the Min-max Procedure can be solved by the "threshold method" of O. Gross [6] (see also [5] and [1]). In the n×n array associated with the (bottleneck) assignment problem, call a row or column a <u>line</u>. Call a set S of cells (elements of the array) <u>independent</u>, if no two cells of S are in the same line. An independent set of cardinality n is an <u>assignment</u>. The "threshold method" starts with some initial assignment S_o, defines the threshold

$$\theta = \max_{i,j}\{d_{ij} \mid (i,j) \in S_o\}$$

and declares <u>admissible</u> those cells (i,j) such that $d_{ij} < \theta$. It then uses a labeling procedure to find a maximal independent set S of admissible cells. If $|S| < n$, then S_o defines an optimal solution x^o to (BAP), with $x_{ij} = 1$ if $(i,j) \in S_o$, $x^o_{ij} = 0$ otherwise. If $|S| = n$, the threshold can be lowered: one sets $S_o \leftarrow S$, redefines θ, and repeats the procedure.

We have slightly modified this basic procedure in order to use the fact that we are solving a sequence of interrelated bottleneck assignment problems rather than just one. The main change is that, rather than lowering the threshold each time to the value of the next largest element of D, we calculate an upper and a lower bound on the objective function value and set the threshold via bisection of the interval defined by these bounds. This has speeded up the procedure considerably.

To generate a starting assignment, we use a "stingy" heuristic that chooses admissible cells in order of increasing d_{ij}. If we have to stop short of choosing n cells, we complete the assignment via the labeling procedure. The details follow.

A line of the assignment array will be called <u>free</u> if it has no cell in the current assignment. At the start D is the traffic matrix, subsequently D is updated at the end of each iteration. Set k = 0 and go to the

<u>Iterative Step</u>

1. (Construct Initial Assignment)

If k = 0, set $\theta = 0$ and go to 1.2. Otherwise let D^k be the last switch matrix produced; set $\theta = \max_{i,j} d^k_{ij}$, declare admissible all cells (i,j) such that $d_{ij} < \theta$, and go to 1.1.

1.1. Scan the rows and put into S the first admissible cell in a free column encountered in each row. If $|S| = n$ go to 2, else continue.

1.2. Find $d_{i_o j_o} = \min\{d_{ij} | i \text{ and } j \text{ are free}\}$. If $d_{i_o j_o} < \infty$, put (i_o, j_o) into S and go to 1.2; else continue.

1.3. If $|S| = n$, go to 2; else declare all cells admissible and complete the assignment by the labeling procedure: set $\delta(\theta) = 0$ and go to 3.

2. (Set New Threshold)

Set $S^* \leftarrow S$, store the candidate assignment S^*, and define

$$\bar{\theta} = \max\{d_{ij} \mid (i,j) \in S\},$$

$$\underline{\theta} = \max\{\theta, \min_{i,j} d_{ij}\},$$

and

$$\delta(\theta) = \begin{cases} 1 & \text{if } \bar{\theta} - \underline{\theta} \leq 1 \\ 0 & \text{otherwise.} \end{cases}$$

Then set

$$\theta \leftarrow \lfloor \tfrac{1}{2}(\bar{\theta} + \underline{\theta}) \rfloor,$$

$$S \leftarrow S \setminus \{(i,j) \in S \mid d_{ij} \geq \theta\},$$

declare admissible all cells (i,j) such that $d_{ij} < \theta$, and go to 3.

3. (Find a Maximal Independent Set of Admissible Cells)

Start with all lines unlabeled, unscanned.

3.1-3.3. (Labeling)

3.1. Label all free rows with -1 and go to 3.2.

3.2. Choose a labeled unscanned row i and scan it for admissible cells. For each such cell (i,j), if column j is unlabeled, label it with i. Then if column j is free, let $j_o \leftarrow j$ and go to 3.4, else continue.

Repeat 3.2 until all labeled rows have been scanned. If no new column has been labeled, go to 3.5; else go to 3.3.

3.3. Choose a labeled unscanned column j and scan it for some $(i,j) \in S$. If such (i,j) is found, label row i with j.

Repeat 3.3 until all labeled columns have been scanned. If no new row has been labeled, go to 3.5; else go to 3.2.

3.4. (Augmentation) If the label of column j_o is i, put (i,j_o) into S. If the label of row i is $j = -1$, make all lines unlabeled, unscanned, and go to 3.1. If the label of row i is $j \neq -1$, put (i,j) into S, set $j_o \leftarrow j$ and go to 3.4.

3.5. S is a maximal independent set of admissible cells.

If $|S| = n$ and $\delta(\theta) = 0$, the threshold can be lowered: go to 2.

If $|S| = n$ and $\delta(\theta) = 1$, S is a min-max assignment: set $S^* \leftarrow S$ and go to 4.

If $|S| < n$, no assignment exists that satisfies the current threshold. Therefore S*, the candidate stored in step 2, is a min-max assignment: go to 4.

4. (Construct New Switch Matrix)

Define $D^{k+1} = (d_{ij}^{k+1})$, where

$$d_{ij}^{k+1} = \begin{cases} d_{ij} & (i,j) \in S^* \\ 0 & (i,j) \notin S^*, \end{cases}$$

and set $d_{ij} \leftarrow \infty$ if $(i,j) \in S^*$, $d_{ij} \leftarrow d_{ij}$ otherwise. If $k + 1 = n$, stop: $(D^1,\ldots,D^n)$ is the desired schedule. Otherwise set $k \leftarrow k + 1$ and go to 1.

A few comments are in order. We will call an application of the Iterative Step a cycle. At the beginning of each cycle, the construction of an initial assignment starts by putting into S only cells that are admissible with respect to the threshold value of the previous iteration (Step 1.1). Step 1.2 continues the construction by including additional cells according to the "stingy" rule of choosing them in order of increasing d_{ij}. When no complete assignment can be obtained in this way, the labeling procedure of Step 3 is used to finish the job, with no restriction on admissibility.

Step 2 is entered with a complete assignment S. Clearly, $\max\{d_{ij} | (i,j) \in S\}$ constitutes an upper bound on the value of the bottleneck objective function for the switch matrix under construction, and $\max\{\theta, \min_{i,j} d_{ij}\}$, where θ is the threshold value of the previous iteration, constitutes a lower bound. To see this latter point, note that if a switch matrix with an objective function value lower than θ existed, it would have been found at the previous iteration. A new threshold θ is then defined by bisecting the interval between the above upper and lower bounds. The role of the parameter $\delta(\theta)$ is to carry to Step 4 the instruction of either continuing the bisection procedure, or terminating it.

Step 3 is the well-known labeling procedure of Ford and Fulkerson for finding a maximal independent set of admissible cells. Steps 2 and 3 are iterated as long as feasible assignments can be found for successively lower thresholds. This process stops when lowering the threshold is no longer possible. At that point the current assignment S* is used to construct the next switch matrix D^{k+1}, the traffic matrix D is updated, and a new cycle is started. After a total of n cycles, the procedure ends with a feasible schedule $(D^1,\ldots,D^q)$, $q = n$.

Next we address the computational complexity of the algorithm. During every cycle, Step 1 is used exactly once, and its time complexity is $O(n^3)$. Step 2 is $O(n^2)$. The labeling procedure of Step 3 is $O(n^2)$, each augmentation is $O(n)$, and the number of augmentations (and hence labelings) during one application of Step 3 is $O(n)$. Hence one application of Step 3 has a total time complexity of $O(n^3)$. Finally, Steps 2 and 3 can be iterated during a cycle at most $\log_2\Delta$ times, where $\Delta = \max_{i,j} d_{ij} - \min_{i,j} d_{ij}$. Thus the time complexity of a cycle is $O(n^3 \log_2 \Delta)$, and since there are n cycles, the time complexity of the whole Min-max procedure is $O(n^4 \log_2 \Delta)$.

Lawler [11] gives a version of the bottleneck assignment algorithm whose time complexity is $O(n^3)$; so when that algorithm is substituted for Steps 2-3 above, the resulting procedure is of $O(n^4)$. We have implemented and extensively tested both procedures. In spite of its better worst case bound, the version using Lawler's algorithm was on the average about 30% slower than the Min-max Procedure using the Ford-Fulkerson method combined with bisection.

3. Computational Results

Two kinds of computational experiments were run. First, both the CMT method [2] and the Min-max Procedure were implemented on an Atari 800 computer, and applied to 200 problems with traffic matrices D of order n = 5, 10, 15 and 20, with 50 problems in each of the 4 classes, and with the elements of D randomly drawn integers from a uniform distribution over the interval [1,100]. Table 1 shows the results.

Table 1. Computational Results on an Atari 800

(50 problems in each class)

n	Average efficiency (%)		Average time (minutes)	
	CMT	Min-max	CMT	Min-max
5	91.28	93.13	0.476	0.583
10	87.65	92.20	7.446	3.016
15	87.35	93.13	42.91	9.043
20	88.62	93.98	149.2	18.04

While the computing time for the CMT method increases more or less proportionally with n^4, i.e., is close to its worst case bound, the time used by the Min-max Procedure grows only slightly faster than n^3, hence is on the average considerably below its worst case bound.

The second computational experiment was run on a VAX computer, and involved randomly generated problems with the elements of D again drawn from a uniform distribution over [1,100]. This time 12 classes of problems were solved, with 1000 problems in each of the first 6 classes (for n = 5, 10, 15, 20, 30 and 40), and 20 problems in each of the next 6 classes (for n = 50, 60, 70, 80, 90 and 100). Table 2 summarizes the results.

Table 2. Computational Results on a VAX 811

(1000 problems in each of the first 6 classes,
20 problems in each of the last 6 classes)

n	Average efficiency (%)	
	CMT	Min-max
5	91.46	93.58
10	88.57	92.73
15	87.62	92.71
20	90.78	95.12
30	88.63	94.81
40	89.57	96.07
50	91.22	97.39
60	90.37	97.69
70	90.27	97.53
80	90.89	97.44
90	91.21	98.28
100	91.48	98.25

The performance of the Min-max Procedure in terms of transmission efficiency is uniformly better than that of the earlier CMT heuristic, and increasingly so for large problems. While the efficiency of the CMT method fluctuates around 89-91% for small as well as for large problems, the efficiency of the Min-max Procedure is around 93% in the range $5 \leq n \leq 15$, around 95% for $20 \leq n \leq 40$, around 97.5% for $50 \leq n \leq 80$, and above 98% for $90 \leq n \leq 100$.

With efficiencies so close to 1, it seems that restricting the number of switch matrices to n is not such a debilitating constraint as it had been thought to be. In the light of this computational experience, Problem 2 of Section 1, first formulated by CMT [2], appears to be the right problem to address, and the Min-max Procedure recommends itself as the method of choice.

The Min-max Procedure, just like the CMT method, can be modified to solve Problems 3 and 4 stated in Section 1. We have tested such modified versions of our procedure and obtained for Problem 3 average transmission efficiencies of 97.86% for $q = 2n$, 98.58% for $q = 3n$ and 99.12% for $q = 5n$, on 100 problems with $n = 20$. These are better than the 95.12% efficiency obtained for Problem 2 (i.e., for $q = n$), but we find it doubtful whether this further improvement can compensate for the disadvantage of having burst splittings.

In the case when proper burst splittings are forbidden (Problem 4), we managed to obtain only a very slight improvement over the efficiency registered for Problem 2 (95.14% for $q = 2n$, 95.39% for $q = 3n$, and 95.83% for $q = 5n$), which seems to rule out Problem 4 as a promising formulation.

Finally, returning to Problem 2, we have also tested a procedure that combines the Min-max heuristic with a limited amount of implicit enumeration in order to find better solutions at increased computational cost. To be more precise, this procedure generates the n alternative solutions obtainable by banning from D^1 (the first switch matrix generated by the Min-max heuristic) one of its n positive entries, and then applying the procedure as usual. At the cost of an n-fold increase in computing time, the average improvement in efficiency was 0.21%, 0.28%, 0.22% and 0.18% for the 50 problems with n = 5, 10, 15 and 20 respectively. These meager improvements strongly suggest that the solutions to Problem 2 obtained by the Min-max Procedure are very close to the optimum.

References

[1] H. Arnold, "An Efficient Digital Satellite Technique for Serving Users of Differing Capacities." International Conference on Communication, Chicago, 1977, p. 6.1-116-119.

[2] P. Camerini, F. Maffioli and P. Tartara, "Some Scheduling Algorithms for SS/TDMA Systems." Fifth International Conference on Telecommunications, Genoa, 1981, p. 405-409.

[3] N. Christofides, Graph Theory: An Algorithmic Approach. J. Wiley, 1975.

[4] V. Chvatal, "A Greedy Heuristic for the Set Covering Problem." Mathematics of Operations Research, 4, 1979, p. 233-235.

[5] L. R. Ford and D. R. Fulkerson, Flows in Networks. Princeton University Press, 1962.

[6] O. Gross, "The Bottleneck Assignment Problem: An Algorithm." Proceedings of the RAND Symposium on Mathematical Programming, RAND Publication R-351, 1960, p. 87-88.

[7] P. Hall, "On the Representation of Subsets." Journal of the London Mathematical Society, 10, 1935, p. 26-30.

[8] T. Inukai, "An Efficient SS/TDMA Time Slot Assignment Algorithm." IEEE Transactions on Communications, COM 27, 1979, p. 1449-1455.

[9] Y. Ito, Y. Urano, T. Muratani, M. Yamaguchi, "Analysis of a Switch Matrix for an SS/TDMA System." Proceedings of theIEEE, 65, 1977, p. 411-419.

[10] H. W. Kuhn, "The Hungarian Method for the Assignment Problem." Naval Research Logistics Quarterly, 2, 1955, p. 83-97.

[11] E. L. Lawler, Combinatorial Optimization: Networks and Matroids. Holt, Rinehart and Winston, 1976.

[12] W. Schmidt, "An On-Board Switched Multiple Access System for Millimeter Wave Satellites." International Conference on Digital Satellite Communication, London, 1969, p. 399-406.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: MSRR 491
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): Traffic Assignment in Communication Satellites
5. TYPE OF REPORT & PERIOD COVERED: Technical Report, April 1983
6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): Egon Balas and Philip R. Landweer
8. CONTRACT OR GRANT NUMBER(s): N00014-82-K-0329, NR 047-607

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA 15213
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS: Personnel and Training Research Programs, Office of Naval Research (Code 434), Arlington, VA 22217
12. REPORT DATE: April 1983
13. NUMBER OF PAGES: 17

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)
15. SECURITY CLASS. (of this report): Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Traffic Assignment, Communication Satellites, SS/TDMA System, Scheduling

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A high capacity communication satellite interconnects scores of ground stations simultaneously. Under the Satellite-Switched/Time Division Multiple Access (SS/TDMA) system, each channel of the satellite is allocated to a pair of ground stations for a certain time period, after which the whole set of allocations (called a switch) is changed simultaneously. The problem we address is to minimize the time length of the entire sequence of switches, subject to a limit on the number of switches. We formulate this as a 3-index bottleneck-sum assignment problem, and solve it by a heuristic that obtains consistently better results than earlier methods based on different formulations.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)


END
FILMED